**Methodology/Technique for Mira Virtual Assistant**

**1. Planning**

* **Goal**: Create a virtual assistant that can respond to user input and provide helpful information like time, or answers to basic queries.
* **Tools**: Use **HTML** for structure, **CSS** for design, and **JavaScript** for functionality.

**2. Design**

* **User Interface (UI)**:
  + Create a webpage that includes:
    - A **display area** where responses will appear.
  + Add simple styling using **CSS** to make the page visually appealing.
  + Ensure the design is **responsive** so it works on all devices (desktop, mobile, tablet).

**3. Implementation**

* **Core Functionalities**:
  1. **Handle User Input**:
     + Use JavaScript to capture voice by the user.
  2. **Keyword Matching**:
     + Create a simple logic to match keywords like:
       - "time" → Display the current time.
       - "hello" → Respond with a friendly greeting.
  3. **Display Responses**:
     + Use JavaScript to dynamically show the assistant’s response in the display area.
  4. **Voice Features (Optional)**:
     + Use the **Web Speech API** to allow voice commands and text-to-speech responses.
* **API Integration (Optional)**:
  1. Use simple APIs for live data:
     + **Time Functions** in JavaScript to show the current time.

**4. Testing**

* Test the project to ensure it works as expected:
  + Input different queries to check if the assistant responds correctly.
  + Test on various devices and browsers to ensure compatibility.
  + Debug any issues (e.g., incorrect responses, layout problems).

**5. Deployment**

* Host the project online for easy sharing:
  + Use **GitHub Pages**, **Netlify**, or any free hosting platform.
  + Share the hosted link with users for feedback.

**Techniques Used**

1. Ensure the UI works seamlessly across all screen sizes using CSS. **Keyword Matching**:
   * Use JavaScript to detect specific words or phrases in user input and trigger predefined responses.
2. **Event Handling**:
   * Capture user actions, such as clicking the submit button or pressing "Enter."
3. **API Integration** (Optional):
   * Use APIs to provide dynamic and real-time responses (e.g., weather data).
4. **Speech Recognition** :
   * Use browser-based **Web Speech API** for voice input and text-to-speech output.